IN THE SPECIFICATION

Please replace the following paragraph beginning on page 6, line 3, with the following rewritten paragraph:

For a more complete understanding of the present invention, and the advantages thereof, reference is now made to the following descriptions taken in conjunction with the accompanying drawings, in which:

FIGURE 1A is a high level functional block diagram of a representative data processing system suitable for practicing the principles of the present invention;

FIGURE 1B is a high level functional block diagram of selected operational blocks within a CPU;

FIGURE 2A illustrates operation tracking queue and link stack mechanisms;

FIGURE 2B illustrates phases of the life cycle of operations used in embodiments of the present invention;

~~FIGURES 2C-2D illustrate an operation tracking queue and a link stack;~~

FIGURE 2C illustrates an operation tracking queue and a link stack;

FIGURE 2D illustrates another operation tracking queue and a link stack;

~~FIGURES 2E-2G illustrate examples of operation states in an operation tracking queue and a link stack;~~

FIGURE 2E illustrates an example of operation states in an operation tracking queue and a link stack;

FIGURE 2F illustrates another example of operation states in an operation tracking queue and a link stack;

FIGURE 2G illustrates another example of operation states in an operation tracking queue and a link stack;

FIGURE 3A illustrates operation tracking queue and link stack mechanisms in an embodiment of the present invention;

~~FIGURES 3B-3C illustrate examples of operation tracking queue and link stack operation algorithms used in embodiments of the present invention;~~

FIGURES 3B illustrates an example of operation tracking queue and link stack operation algorithms used in embodiments of the present invention;

FIGURES 3C illustrates an example of operation tracking queue and link stack operation algorithms used in embodiments of the present invention;

~~FIGURES 3D-3F illustrate examples of operation tracking queue and link stack operation in embodiments of the present invention;~~

FIGURES 3D illustrates an example of operation tracking queue and link stack operation in embodiments of the present invention;

FIGURES 3E illustrates another example of operation tracking queue and link stack operation in embodiments of the present invention;

FIGURES 3F illustrates another example of operation tracking queue and link stack operation in embodiments of the present invention;

FIGURE 4A illustrates an operation tracking queue and link stack used in alternate embodiments of the present invention;

~~FIGURES 4B-4D illustrate an operation tracking queue and link stack algorithms used in alternate embodiments of the present invention; and~~

FIGURES 4B illustrates an operation tracking queue and link stack algorithms used in alternate embodiments of the present invention;

FIGURES 4C illustrates another operation tracking queue and link stack algorithms used in alternate embodiments of the present invention;

FIGURES 4D illustrates another operation tracking queue and link stack algorithms used in alternate embodiments of the present invention;

~~FIGURES 4E-4G illustrate examples of an operation tracking queue and link stack operation in alternate embodiments of the present invention.~~

FIGURES 4E illustrates an example of an operation tracking queue and link stack operation in alternate embodiments of the present invention;

FIGURES 4F illustrates another example of an operation tracking queue and link stack operation in alternate embodiments of the present invention; and

FIGURES 4G illustrates another example of an operation tracking queue and link stack operation in alternate embodiments of the present invention.

Please replace the following paragraph beginning on page 16, line 4, with the following rewritten paragraph:

FIGURE 2G illustrates register states of OTQ 201 and LS 211 in the case where one entry (PUSH IJKL) has been flushed. Register 223 contains the operation that is to be flushed (by definition for this example); this determination was made by other processor logic (not shown). ALLOCATE_ptr 205 points to a blank register 223. Since only one operation is to be flushed, execution of OTQ_FLUSH 279 places FLUSH_ptr 208 to the desired point in OTQ 201 (register 223). The number of operations to be flushed is calculated as in step 281 of FIGURE 2D, then ALLOCATE_ptr 205 is set to the value of FLUSH_ptr 208 (points to register 223) and then the FLUSH_ptr 208 is decremented to point to register 216. ALLOCATE_ptr 205 is now pointing to the next register position in which a new operation will be allocated (register 223). Register 223 contains the operation that has been flushed (a new allocated operation will write over the information in register 223). Step 284 of OTQ_FLUSH 279 in FIGURE [[2D]] 2E sets CUR_LS_ptr 213 to the value found in the LS_ptr field 203 of the register pointed to by FLUSH_ptr 208 (register 216 with LS_ptr field value of 3) once it has been decremented. LS 211 register 225 (indicated by 3) contains an address 215 (IJKL). The method in FIGS. 2C-2D avoids the flushed address IJKL by decrementing CUR_LS_ptr 213 before a POP address is read from the link stack LS 211.

Please replace the following paragraph beginning on page 11, line 19, with the following rewritten paragraph to make the Specification conform to new FIG 2B:

FIGURE 2B is a flow diagram of processor operations that may occur when an OTQ 201 and an LS 211 are used to track operations. When a processor is doing speculative instructive execution (using algorithms to predict and execute future instructions) outside of the "committed" or actual instruction execution stream, then the instructions may trigger operations that are tracked in an OTQ 201 and LS 211. In step [[220]] 230, an instruction is fetched from an instruction cache (e.g., I_CACHE 151). The instruction is decoded in step [[221]] 231. Decoding an instruction leads to a determination whether the instruction should also trigger operations that will be tracked in a queue(e.g., exemplary OTQ 201). If an instruction triggers an operation that is to be tracked, then an OTQ_ALLOC 239 (described in conjunction with the flow diagram in FIGURE 2C) would be used to allocate or add the operation to OTQ 201. Since the processor may have many instructions in its pipeline, there may be many computer cycles between a decode in step [[221]] 231 and an actual instruction execution in step [[223]] 232. An execution in step [[222]] 232 would generate results that would be compared to results from a corresponding speculative execution. This compare may determine that a speculative (look ahead) path has been actually taken in the committed instruction execution stream. If the results of the instruction execution of step [[222]] 232 determine that a sequence of speculative instructions (operations tracked in OTQ 201) will not be executed in the committed instruction execution stream, then the OTQ 201 operations are flushed or removed. A FLUSH in step [[225]] 235 triggers an OTQ_FLUSH 279 (described in conjunction with the flow diagram in FIGURE 2E) which flushes operations of OTQ 201. If the execution in step [[222]] 232 generates results that indicate that the speculative instructions will be executed, then a commit in step [[224]] <u>234</u> executes an OTQ_DEALLOC 259 (described in conjunction with the flow diagram in FIGURE 2D) which will remove the OTQ 201 operation (pointed to by DEALLOCATE_ptr 206) and possibly an associated address 216 in the LS [[213]] <u>211</u>. It should be noted that the operations in the OTQ 201 and the LS 211 are not removed in the sense of erased, instead removed operations are free to be written over by subsequent OTQ 201 operations.